PATENTTI- JA REKISTERIHALLITUS
NATIONAL BOARD OF PATENTS AND REGISTRATION

Helsinki 15.1.2001

#3

E T U O I K E U S T O D I S T U S
P R I O R I T Y D O C U M E N T

Hakija Applicant	**Nokia Mobile Phones Ltd** **Espoo**
Patenttihakemus nro Patent application no	**20000566**
Tekemispäivä Filing date	**10.03.2000**
Kansainvälinen luokka International class	**H04L**

Keksinnön nimitys
Title of invention

"Sovelluskohtaisen palvelunlaadun optimointien ohjaus"

Täten todistetaan, että oheiset asiakirjat ovat tarkkoja jäljennöksiä patentti- ja rekisterihallitukselle alkuaan annetuista selityksestä, patenttivaatimuksista, tiivistelmästä ja piirustuksista.

This is to certify that the annexed documents are true copies of the description, claims, abstract and drawings originally filed with the Finnish Patent Office.

Pirjo Kaila
Tutkimussihteeri

Maksu 300,- mk
Fee 300,- FIM

Osoite: Arkadiankatu 6 A, P.O.Box 1160, FIN-00101 Helsinki, FINLAND
Puhelin: 09 6939 500
Telephone: + 358 9 6939 500
Telefax: 09 6939 5328
Telefax: + 358 9 6939 5328

h2

Sovelluskohtaisten palvelunlaadun optimointien ohjaus

Nyt esillä oleva keksintö kohdistuu oheisen patenttivaatimuksen 1 johdanto-osan mukaiseen menetelmään, jonka avulla voidaan ohjata sovelluskohtaisia palvelunlaadun optimointeja käyttäen yleisiä palvelutason signalointiprotokollia. Keksintö kohdistuu lisäksi oheisen patenttivaatimuksen 10 johdanto-osan mukaiseen palvelutason signalointiprotokollaan.

Tiedonsiirtoverkoissa, kuten TCP/IP-verkoissa (Transmission Control Protocol/Internet Protocol) on tarjolla erilaisia palvelunlaadun (QoS, Quality of Service) signalointiprotokollia, kuten RSVP (Resource ReSerVation Protocol) ja DiffServ (Differentiated Services). Näitä protokollia käytetään viestittämään tiedonsiirtoverkon solmuille miten tiettyjen pakettien käsittelyssä tulisi toimia. Tämä tarkoittaa esim. sitä, että nämä paketit tulisi reitittää eri reittiä pitkin, että reitittimien pitäisi lajitella nämä paketit niille sopivaan jonoon, tai että nämä paketit tulisi välittää käyttäen erilaista linkkiä tai linkkitason palvelua (esim. ATM:n tapauksessa eri virtuaaliyhteyttä).

Edellä luetellut palvelunlaatuun kohdistuvat operaatiot ovat riippumattomia sovelluksesta, mutta on olemassa myös eri sovelluksille ominaisia eli sovelluskohtaisia palvelunlaatuun vaikuttavia operaatioita. Esimerkkinä voidaan mainita RTP-kohtainen optimointi (Transport Protocol for Real-Time Applications), eli RTP-otsikon pakkaus.

Otsikonpakkauksen lisäksi on mahdollista käyttää useita eri RTP-kohtaisia suorituskyvyn optimointikeinoja. Näiden käyttämisen esteenä on se, että tiedonsiirtoverkon solmun on vaikea tunnistaa RTP-tietovirta (RTP Stream) luotettavasti muusta liikenteestä. Tämä johtuu siitä, että RTP:ssä ei ole olemassa tiettyjä portteja tai luotettavasti tunnistettavia yksilöllisiä otsikkokenttiä. Tämän takia ei ole kannattavaa käynnistää RTP-kohtaista optimointia tietylle tietovirralle. Se voi olla jopa mahdotonta, jos esim. RTP-optimointi muuttaa liikenteen kulkemaa reittiä.

Ongelmana on, että tiedonsiirtoverkon solmujen ei ole mahdollista tunnistaa luotettavasti RTP-paketteja käyttäen yleisiä ja suoraviivaisia keinoja. Yleensä tietyt sovellukset erotetaan muusta liikenteestä siitä,

että ne käyttävät tiettyä kuljetusprotokollan porttia (esim. HTTP käyttää TCP-porttia 80), tai niiden otsikon kentät pystytään tunnistamaan. Vaikka on suositeltu, että sovellukset käyttävät UDP-porttia 5004 RTP-liikenteelle, niin käytännössä RTP:n kanssa käytetään satunnaisesti valittuja porttinumeroita. Tämä johtuu siitä, että yhteen sovellukseen kuuluu yleensä useita eri tietovirtoja, jotka kukin käyttävät eri UDP-porttia, ja lisäksi useat sovellukset voivat yhtäaikaa käyttää RTP:tä samassa tietokoneessa. Samoin RTP-otsikko on hyvin yksinkertainen ja useimmat siinä olevat arvot ovat luonteeltaan satunnaisia.

Eräs tapa RTP-tietovirran tunnistukseen luotettavasti on istuntotason protokollan tutkiminen. Istuntotason protokollassa (esim. H.245 tai SDP) välitetään istuntoon kuuluvien RTP-yhteyksien käyttämät IP-osoitteet ja portit, samoin kuin käytettävät koodekit ja pakettien koot. Ongelmana on, että tällaisen istuntotason protokollan paketit kulkevat tyypillisesti eri reittiä kuin RTP-yhteys, ja että paketit on tyypillisesti salattu päästä päähän.

Tunnetun tekniikan mukaan yleensä voidaan kuitenkin olettaa, että RTP-otsikonpakkausta tai RTP-multipleksausta on mahdollista soveltaa suoraan RTP-lähteessä tai RTP-vastaanottajaa edeltävässä solmussa. Tässä tapauksessa lähde tai vastaanottaja voi käyttää linkkikerroskohtaista (Link Layer) signalointia aktivoidakseen otsikonpakkauksen (Header Compression) tai multipleksauksen toisessa päässä yhteyttä. On mahdollista levittää tiedonsiirtoverkkoon tieto siitä, että jos vastaanotettavassa tietovirrassa on käytössä otsikonpakkaus, niin tiedonsiirtoverkon solmu voi käyttää otsikonpakkausta myös samaan tietovirtaan sen lähtiessä toista linkkiä pitkin.

Jos sovellus ei pysty signaloimaan RTP-tietovirtaa tiedonsiirtoverkkoon, vaan pystyy vain hallitsemaan linkkiä, joka on kytketty suoraan tietokoneeseen, jossa sovellusta ajetaan, voidaan RTP-otsikonpakkausta käyttää vain tiedonsiirtoverkon laidoilla. Tunnetun tekniikan mukaisesti RTP-multipleksointia voidaan käyttää vain laitteiden välillä, joiden välillä on useita päästä-päähän-yhteyksiä. Ei ole siis mahdollista käyttää RTP-multipleksointia esim. kahden reitittimen välillä, joiden välillä kulkee useita RTP-tietovirtoja.

Keksinnön eräänä tarkoituksena on aikaansaada menetelmä, jonka avulla voidaan käynnistää sovellukselle ominaisia liikenteen optimointeja lähettäjän ja kohteen välisissä tiedonsiirtoverkon solmuissa, kuten reitittimissä.

Tämä tarkoitus voidaan saavuttaa siten, että sovellus käyttää palvelutason (QoS, Quality of Service) signalointiprotokollia merkitsemään sovelluskohtaisia tietovirtoja. Tällöin tiedonsiirtoverkon solmut voivat palvelutason signaloinnin perusteella tunnistaa sovelluksen tietovirrat, jolloin sovelluskohtainen optimointi, kuten RTP-otsikonpakkaus tai -multipleksaus, on mahdollista käynnistää, jopa ennen kuin sovellus on lähettänyt mitään varsinaista liikennettä.

Täsmällisemmin sanottuna keksinnön mukaiselle menetelmälle on tunnusomaista se, mikä on esitetty patenttivaatimuksen 1 tunnusmerkkiosassa. Keksinnön mukaiselle palvelutason signalointiprotokollalle on tunnusomaista se, mikä on esitetty patenttivaatimuksen 10 tunnusmerkkiosassa.

Nyt esillä olevalla keksinnöllä saavutetaan merkittäviä etuja tunnetun tekniikan mukaisiin ratkaisuihin verrattuna. Kun tietovirran (esim. RTP) luonne voidaan signaloida lähettäjän ja kohteen välissä oleville solmuille (esim. reitittimille), mahdollistetaan useita tietovirtakohtaisia optimointeja. Tällainen signalointi mahdollistaa sovelluksille myös standardin rajapinnan (API, Application Programming Interface) näihin optimointeihin. Tällöin sovellusta voidaan käyttää muuttamattomana eri tyyppisissä aliverkoissa, kuten PPP, GPRS, WLAN.

Keksintöä selostetaan seuraavassa tarkemmin viitaten samalla oheisiin piirustuksiin, joissa

kuva 1 esittää periaatteellisena kaaviokuvana tilannetta, jossa ensimmäinen tietokone H1 siirtää RTP-tietovirtaa toiselle tietokoneelle H2 tiedonsiirtoverkon yli,

kuva 2 esittää tiedonsiirtoverkossa kulkevaa RTP-pakettia,

kuva 3 esittää periaatteellisena kaaviokuvana tilannetta, jossa RSVP:tä käytetään RTP:n kanssa, ja

kuva 4 esittää periaatteellisena kaaviokuvana tilannetta, jossa DiffServ:iä käytetään RTP:n kanssa. Eri palveluluokat on merkitty kuvaan merkeillä ♣, ♦, ♥ ja ♠.

RTP on yksinkertainen protokolla, joka lisää hyötykuorman eteen 12 oktettia pitkän RTP-otsikon 12. Tämä otsikko, joka on esitetty kuvassa 2, käsittää mm. versionumeron 14, hyötykuorman tyypin tunnisteen 15 (PT), sarjanumeron 16 (Sequency number), aikaleiman 17 (Timestamp) ja lähteen tunnisteen 18 (Synchronization Source Identifer). Lähde- tai kohdeosoitteita ei ole, joten osoitukseen käyttetään RTP:n alapuolella olevaa kuljetusprotokollaa, kuten UDP (User Datagram Protocol), TCP (Transmission Control Protocol) tai ATM-AAL5 (Asynchronous Transfer Mode Adaptation Layer type 5).

RTP:n toimintaa tukemaan käytetään tavallisesti RTCP-ohjausprotokollaa (Real-Time Transport Control Protocol). RTCP:tä käytetään RTP-tietovirtojen synkronointiin, sekä viiveen vaihtelun ja pakettihukan valvontaan. RTCP-liikenne koostuu tyypillisesti lähetys- ja vastaanottoraporteista. RTP-liikennettä ja siihen liittyvää RTCP-liikennettä kutsutaan nimellä RTP/RTCP. Kaikki RTP-sovellukset eivät kuitenkaan käytä RTCP:tä, sen käyttö on valinnaista. Lähetettäessä RTP:tä UDP:n päällä RTP-liikenteen porttinumerot ovat parillisia ja RTP:tä tukevan RTCP-liikenteen yhtä suurempia ja parittomia.

Kuvaan 1 viitaten ensimmäinen tietokone 1a siirtää RTP-tietovirtaa toiselle tietokoneelle 1b. Itse siirtoprosessi on melko yksinkertainen. Reaaliaikasovellus saa hyötykuorman 13 koodekilta (ei esitetty kuvassa), varustaa sen RTP-otsikolla 12, joka käsittää mm. nykyisen aikaleiman 17, järjestysnumeron 16 ja lähteen tunnisteen 15 (32-bittinen satunnaisluku). Sen jälkeen, kun edellä mainittu sovellus toimittaa paketin TCP/IP:lle, TCP/IP varustaa saadun paketin UDP- ja IP-otsikoilla ja lähettää tämän toiselle tietokoneelle 1b. Kuvassa 2 on esitetty tämä lähetettävä paketti 3, jossa on IP-otsikko 10, UDP-otsikko 11, RTP-otsikko 12, hyötykuorma 13 (Media payload) ja mahdollisesti täyteoktetteja 14 (Pad).

Hyötykuorma 13, jota välitetään RTP:n yli, voi olla hyvin lyhyt. Tällaisissa tapauksissa RTP-, UDP- ja IP-otsikot 10, 11 ja 12 edustavat suurta osaa koko paketin 3 koosta, esim. 40 tavua IPv4 tapauksessa.

RTP-otsikonpakkausta voidaan käyttää esim. silloin, kun käytetään hitaan nopeuden linkkiä. RTP-otsikonpakkaus ei siirrä otsikoiden vakioosia, kuten lähteen osoitetta 19 (Source IP address) ja kohteen osoitetta 20 (Destination IP address) sen jälkeen, kun pakattu sisältö on saatu valmiiksi. Vaihtuvat kentät, kuten RTP-aikaleima 17, järjestysnumero 16 tai IP-tunniste 21 (Identification) pakataan joko lähettämällä peräkkäisten arvojen erotus tai käyttämällä hyväksi tietoa, että tämä ero on vakio.

Jotkut linkkikerroksen toteutukset määräävät minimikoon paketille, esim. 64 oktettia Ethernetissä ja 512 oktettia Gigabit Ethernetissä, joten otsikonpakkaus ei välttämättä yksistään riitä. Yleensä pakettikytkentäisissä tiedonsiirtoverkoissa kunkin paketin prosessointiin käytetään tietty vakioaika, jolloin pakettien lukumäärän vähentäminen parantaa tiedonsiirtoverkon suorituskykyä. Tässä tapauksessa sopiva optimointitapa on multipleksata useita RTP-yhteyksiä yhteen verkkokerroksen yhteyteen. Kun useita RTP-paketteja voidaan välittää yhdessä verkkokerroksen paketissa 3, tiedonsiirtoverkon kuorma kevenee ja lisäksi otsikoiden osuus koko liikenteestä vähenee. Multipleksatun yhteyden päätepisteet neuvottelevat multipleksattujen yhteyksien parametrit siten, että alkuperäisen kaltaiset RTP-tietovirrat voidaan aikaansaada uudelleen multipleksauksen purkamisen jälkeen.

Tietovirta, joka siirretään RTP:n yli, tarvitsee yleensä tiedonsiirtoverkolta erilaista suorituskykyä kuin muu liikenne. RTP-tietovirta on huomattavasti herkempi hukkuneille paketeille ja viiveelle kuin tavallinen, esim. TCP:tä käyttävä liikenne. Riittävän palvelutason saavuttamiseksi olisi hyvä, että RTP:tä käyttävä sovellus käyttäisi myös tiedonsiirtoverkon tarjoamia palvelutasomekanismeja. Näiden palvelutasomekanismien käyttäminen vaatii jonkinlaista signalointia, sovelluksen on mm. informoitava tiedonsiirtoverkkoa siitä, mille paketeille annetaan etuoikeus. Signalointiin voidaan käyttää esim. RSVP:tä tai DiffServin TOS-oktettia.

Palvelutason signaloinnin seurauksena tiedonsiirtoverkon välisolmut 2a, 2b, 2c ja 2d (kuva 1) voivat kohdella tietovirtaan kuuluvia paketteja 3 halutulla tavalla. Välisolmut voivat tunnistaa tietovirtaan kuuluvat paketit, puskuroida niitä ja siirtää niitä ottaen huomioon palvelutason vaatimukset, jotka sovellus on asettanut palvelutason signaloinnin avulla.

Palvelutason signalointi voi myös käynnistää RTP-kohtaisen optimoinnin, kuten otsikonpakkauksen tai multipleksauksen. Käynnistäminen voi olla joko eksplisiittistä tai implisiittistä. Eksplisiittisessä tapauksessa tietty palveluluokka tai tietovirran määrittely koskee vain RTP-liikennettä. Implisiittisessä tapauksessa välisolmut 2a, 2b, 2c, 2d voivat käyttää palvelutason signalointia lisätietona päätettäessä sovelletaanko optimointia tiettyyn tietovirtaan vai ei.

Keksintö ei rajoitu mihinkään tiettyyn signalointiprotokollaan, vaan sitä voidaan hyödyntää minkä tahansa palvelutason signaloinnin kanssa. Seuraavassa keksintöä kuvataan käyttämällä esimerkkeinä kahta yleisintä palvelutason signalointiprotokollaa: RSVP:tä ja DiffServ:iä. Nämä protokollat on valittu esimerkeiksi, koska ne edustavat kahta eri tyyppistä signalointia. Tällä halutaan myös korostaa sitä, että keksintöä voidaan soveltaa monien erityyppisten signalointiprotokollien tapauksessa.

RSVP:ssä palvelun varaus tapahtuu seuraavan selostuksen mukaisesti viitaten kuvaan 3. Lähdesolmu 1a lähettää *Path*-viestin 4 alavirtaan kohti kohdetta 1b, eli RTP-tietovirran 6 suuntaan. Kaikki lähteen ja kohteen välissä olevat solmut 2a, 2b, 2c, 2d, jotka pystyvät käsittelemään RSVP:tä, ottavat tämän *Path*-viestin 4 vastaan, avaavat tietovirralle polun, lisäävät oman osoitteensa viestiin ja välittävät viestin eteenpäin kohti kohdetta 1b.

Lähde 1a kuvaa tietovirran ominaisuudet *Path*-viestissä 3 olevalla *Tspec*-objektilla, joka määrittelee tietovirran normaalin siirtonopeuden (tavuina sekunnissa) ja purskeisuuden (maksiminopeus tavuina sekunnissa ja puskurin koko tavuina). Viestissä olevassa *SENDER_TEMPLATE*-objektissa annetaan tietovirran lähettäjän osoite, ja *SESSION*-objektissa virran kohteen osoite. . Lähteen 1a ja kohteen

1b välissä olevat solmut 2a, 2b, 2c, 2d voivat lisätä palvelutason resurssien määrittelyjä ja käytettävissä olevat ominaisuudet *Path*-viestissä olevaan *ADspec*-objektiin. Vastaanottajalle 1b saapuessaan *Path*-viesti sisältää siis tietovirran kuvauksen (*Tspec*) ja kuvauksen reitin varrella olevista palvelutason resursseista (*ADspec*).

Vastaanotettuaan *Path*-viestin 4 vastaanottava tietokone eli kohdesolmu 1b voi varata palvelutason resurssit. Jotta tämä onnistuisi, kohde lähettää *Resv*-viestin 5 takaisin ylävirtaan, lähdettä 1a kohden. Tämä *Resv*-viesti käsittää kuvauksen siitä, minkälaista palvelua vastaanottaja haluaa. Haluttu palvelu kuvataan *Resv*-viestissä olevassa *flowspec*-objektissa, joka muodostuu *Tspec*- ja *Rspec*-objekteista. Kukin välisolmu 2a, 2b, 2c, 2d varaa tämän perusteella halutut resurssit ja edelleenlähettää viestin ylävirtaan (kohti lähettäjää) *Path*-viestissä 4 olleen osoitteen perusteella. Kun varaus on tehty koko reitin varrella, lähde 1a voi lähettää kuittauksen varauksesta *ResvConf*-viestillä (ei esitetty). Normaalisti *ResvConf*-viesti lähetetään virran kohteelle 1b, jotta kohde tietäisi, että varaus ollaan suoritettu. Tämän jälkeen kun paketti 3 (kuva 1) saapuu johonkin tiedonsiirtoverkon solmuun, tämän paketin sisältämän tiedon tyyppi voidaan tunnistaa, jolloin tähän pakettiin voidaan kohdistaa tälle tyypille ominaisia optimointikeinoja.

RSVP on oliokeskeinen, millä tarkoitetaan sitä, että kaikkien RSVP-viestejä käsittelevien solmujen 2a, 2b, 2c, 2d ei tarvitse ymmärtää ja käsitellä kaikkia viesteissä olevia kenttiä. Solmut, jotka eivät osaa käsitellä joitain kenttiä, vain välittävät ne eteenpäin. RSVP:n kentät eli oliot ovat rakennettu siten, että niitä voidaan helposti laajentaa. On mahdollista määritellä uusia palveluja, jotka signaloidaan käyttäen RSVP:tä muuttamatta protokollan perusrakennetta tai olemassa olevien palveluiden (mm. controlled load service, guaranteed service) toteutusta niissä solmuissa, jotka eivät tue uusia palveluja.

RSVP:hen tehtävät laajennukset on esitetty seuraavassa:

- Lähde1a ilmaisee ehdottamansa tietovirran olevan RTP/RTCP:tä *Tspec*:issä. Tätä tarvitaan esim. silloin, kun liikenteen kulkema polku muuttuu tunneloinnin seurauksena multipleksatuilla yhteyksillä.

- Lähettäjän 1a ja kohteen 1b välissä olevat solmut 2a, 2b, 2c, 2d voivat ilmaista millaista RTP/RTCP-kohtaista palvelutason optimointia ne pystyvät toteuttamaan *ADspec*:issä.
- Kohde 1b ilmaisee tietovirran olevan RTP/RTCP:tä *flowspec*:issä.

Helpoin tapa toteuttaa RTP-tuki on lisätä uudet sovelluskohtaiset palveluparametrit edullisesti olemassa olevaan palveluiden kuvaukseen. Sovelluskohtaiset palveluparametrit voivat sisältää lippuja, jotka ilmaisevat millaista käsittelyä kohde 1b tarvitsee, jotta se pystyisi ottamaan vastaan tietovirtaa onnistuneesti. Esimerkiksi on mahdollista ottaa käyttöön IP-otsikonpakkaus, UDP-otsikonpakkaus ja RTP-otsikonpakkaus. Sovelluskohtaiset palveluparametrit voivat myös sisältää tarvittavan tiedon herättääkseen optimoinnin ennen kuin mitään liikennettä on edes otettu vastaan.

Seuraavassa käsitellään keksintöä DiffServ:in tapauksessa viitaten samalla kuvaan 4. DiffServ:issä ei ole erillisiä signalointiviestejä, vaan toivotut palveluluokat on ilmoitettu erilaisilla merkeillä DS-kentässä (DS-kenttänä käytetään IPv4:ssä TOS-oktettia 22 kuvassa 2) IP-otsikossa 10. Signalointiviesti 9 kulkee itse paketin 3 mukana. Palveluluokat ovat voimassa vain tietyn toimialueen 8a, 8b, 8c sisällä. Rajasolmu 7, joka on kahden eri toimialueen 8a, 8b rajalla, lajittelee saapuvat paketit eri palveluluokkiin ja merkkaa ne kutakin palveluluokkaa vastaavalla merkillä (code point). Tämä lajittelu voidaan suorittaa monella eri perusteella tässä rajasolmussa. Sovellus voi käyttää jotain signalointiprotokollaa esim. RSVP:tä antaakseen tarvittavat lajitteluparametrit rajasolmulle tai sovellus voi itse käyttää esim. IPv4:n tapauksessa TOS-oktettia 22 (Type of service) merkatakseen RTP-paketit 3. Tämän avulla rajasolmu pystyy luotettavasti tunnistamaan RTP-paketit ja merkkaamaan ne sopivalla arvolla DS-kentässä. Kun jokin tiedonsiirtoverkon muista solmuista 2a, 2b, 2c saa RTP:tä käyttävän paketin, se voi turvallisesti käyttää tälle tyypille ominaisia optimointikeinoja.

Kuvassa 4 on esitettynä eräs esimerkki DiffServ:istä. Tässä esimerkissä IP-puhelu käsittää kaksi UDP-tietovirtaa, joista toinen kuljettaa ääntä käyttäen RTP:tä (merkattu kuvaan RTP:nä) ja toinen kuljettaa valkotaulusovelluksen tietoja ilman RTP:tä (merkattu kuvaan WB:nä). Rajasolmu 7 lajittelee RTP-tietovirran RTP-luokkaan ja merkkaa sen

RTP-luokan merkillä (♥). Se lajittelee valkotaulutiedon reaaliaikaluokkaan ja merkkaa sen reaaliaikaluokan merkillä (♠). Muut tiedonsiirtoverkon solmut 2a, 2b, 2c käyttävät RTP-otsikonpakkausta vain tietovirroille, joka on merkitty RTP-luokan merkillä (♥).

Jos käytettävissä ei ole omaa palveluluokkaa juuri RTP:lle, mutta käytössä on joku yleinen reaaliaikapalveluluokka, esim. VoIP (Voice over IP), tiedonsiirtoverkon solmut 2a, 2b, 2c (kuva 4) voivat käyttää palveluluokkaa lisätietona päättäessään käsittääkö tietovirta RTP-liikennettä. Esimerkiksi kuvan 4 tapauksessa, jos ääni olisi merkattu kuuluvaksi reaaliaikapalveluluokkaan, niin tällöin tiedonsiirtoverkon solmut 2a, 2b, 2c voivat tunnistaa äänen RTP-tietovirraksi seuraavilla perusteilla. TOS-oktetin 22 perusteella paketit kuuluvat reaaliaikapalveluluokkaan, lähde- 23 (Source port number) ja kohdeportit 24 (Destination port number) ovat yhdenmukaisia, paketin pituus 25 (Total length) on oleellisesti vakio, RTP versionumero 14 on 2 (V=2), RTP-järjestysnumero 16 (Sequence number) ja RTP-aikaleima 17 (Timestamp) kasvaa monotonisesti ja hyötykuorman tyyppi 15 (PT) ja lähteen tunniste 18 (Synchronization Source Identifier) pysyvät vakioina.

Nyt esillä olevaa keksintöä ei ole rajoitettu ainoastaan edellä esitettyihin suoritusmuotoihin, vaan sitä voidaan muunnella oheisten patenttivaatimusten puitteissa.

43

Patenttivaatimukset:

1. Menetelmä sovelluskohtaisen palvelunlaadun optimoimiseksi, jossa menetelmässä optimoidaan mainitulta sovellukselta tulevien tietovirtojen käsittelyä lähettäjän (1a) ja vastaanottajan (1b) välissä olevissa tiedonsiirtoverkon solmuissa (2a, 2b, 2c, 2d), jossa tiedonsiirtoverkossa käytetään ainakin yhtä palvelutason signalointiprotokollaa, **tunnettu** siitä, että sovellus käyttää mainittua ainakin yhtä palvelutason signalointiprotokollaa merkitsemään sovelluskohtaisia tietovirtoja, jolloin tiedonsiirtoverkon solmut (2a, 2b, 2c, 2d) tunnistavat palvelutason signaloinnin perusteella mainitun sovelluksen tietovirtaan kuuluvat paketit (3) ja niiden tyypin, jolloin näihin paketteihin (3) kohdistetaan tälle tyypille ominaisia optimointikeinoja.

2. Patenttivaatimuksen 1 mukainen menetelmä, jossa menetelmässä tiedonsiirtoverkossa välitetään ainakin yhden tyyppisiä paketteja (3), **tunnettu** siitä, että mainittuna signalointiprotokollana käytetään palvelutason signalointiprotokollaa, joka käsittää pakettien tyypin kuvauksen.

3. Patenttivaatimuksen 1 tai 2 mukainen menetelmä, jossa menetelmässä optimoidaan palvelutasoa, **tunnettu** siitä että signalointiprotokollana käytetään palvelutason signalointiprotokollaa, joka käsittää optimoinneissa tarvittavia parameterejä.

4. Patenttivaatimuksen 1, 2 tai 3 mukainen menetelmä, **tunnettu** siitä, että sovelluskohtaista optimointia käytetään reaaliaikasovelluksen tietovirran optimointiin tiedonsiirtoverkon solmuissa (2a, 2b, 2c, 2d).

5. Patenttivaatimuksen 4 mukainen menetelmä, **tunnettu** siitä, että sovelluskohtaista optimointia käytetään RTP-tietovirran (6) optimointiin.

6. Jonkin patenttivaatimuksen 1-5 mukainen menetelmä, **tunnettu** siitä, että sovellus muodostaa signalointiprotokollan signalointiviestien (4, 5) avulla sovelluksen tietovirtaa (6) varten optimoidun polun lähettäjän (1a) ja vastaanottajan (1b) välille, jolloin jokaiselta mainitun polun varrella olevalta tiedonsiirtoverkon solmulta (2a, 2b, 2c, 2d) varataan sovelluksen tarvitsema optimoitu palvelutaso.

7. Patenttivaatimuksen 6 mukainen menetelmä, **tunnettu** siitä, että signalointiprotokollana käytetään RSVP:tä, jolloin *Path* (4), *Resv* (5) ja *ResvConf* -viestien avulla varataan sovelluksen tietovirralle (6) optimoitu polku lähettäjän (1a) ja vastaanottajan (1b) välille.

8. Jonkin patenttivaatimuksen 1-5 mukainen menetelmä, jossa menetelmässä sovellus lähettää paketteja (3), **tunnettu** siitä, että sovellus liittää lähetettävään pakettiin (3) signalointiviestin (9), jonka perusteella jokainen saavutettu tiedonsiirtoverkon solmu (2a, 2b, 2c), voi suorittaa optimoinnin.

9. Patenttivaatimuksen 8 mukainen menetelmä, **tunnettu** siitä, että signalointiprotokollana käytetään DiffServ:iä, jolloin signalointiviesti (9) kulkee itse paketin (3) mukana paketin DS-kentässä (22) IP-otsikossa (10), jonka avulla kukin saavutettu tiedonsiirtoverkon solmu (2a, 2b, 2c) voi suorittaa optimoinnin.

10. Palvelutason signalointiprotokolla, joka on järjestetty välittämään signalointiviestejä tiedonsiirtoverkon solmuille (2a, 2b, 2c, 2d), ja joka palvelutason signalointiprotokolla käsittää välineet tietyn sovelluksen tietovirran merkitsemiseen, välineet mainitun tietovirran tyypin välittämiseen ja välineet optimointiparametrien välittämiseen, **tunnettu** siitä, että palvelutason signalointiprotokolla on järjestetty merkitsemään mainitulle sovellukselle kuuluvat tietovirrat tiedonsiirtoverkon solmuja (2a, 2b, 2c, 2d) varten, jolloin nämä tiedonsiirtoverkon solmut (2a, 2b, 2c, 2d) on järjestetty tunnistamaan nämä tietovirrat ja käyttämään näihin tietovirtoihin kullekin tyypille ominaisia optimointikeinoja.

4

(57) Tiivistelmä

Keksintö kohdistuu menetelmään sovelluskohtaisen palvelunlaadun optimoimiseksi, jossa menetelmässä optimoidaan mainitulta sovellukselta tulevien tietovirtojen käsittelyä lähettäjän (1a) ja vastaanottajan (1b) välissä olevissa tiedonsiirtoverkon solmuissa (2a, 2b, 2c, 2d). Tässä tiedonsiirtoverkossa käytetään palvelutason signalointiprotokollaa. Sovellus käyttää tätä palvelutason signalointiprotokollaa merkitsemään sovelluskohtaisia tietovirtoja, jolloin tiedonsiirtoverkon solmut (2a, 2b, 2c, 2d) tunnistavat palvelutason signaloinnin perusteella mainitun sovelluksen tietovirtaan kuuluvat paketit (3) ja niiden tyypin, jolloin näihin paketteihin (3) kohdistetaan tälle tyypille ominaisia optimointikeinoja.

Fig. 1

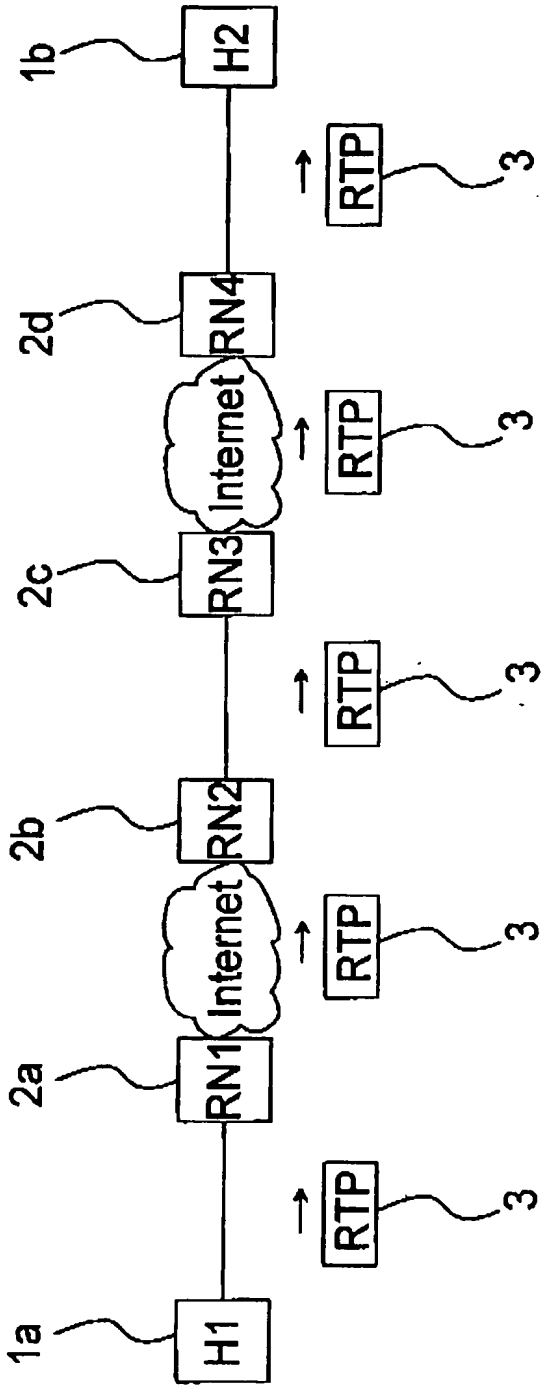
1b
H2
RTP
3
2d
RN4
Internet
RTP
3
2c
RN3
RTP
3
2b
RN2
Internet
RTP
3
2a
RN1
RTP
3
1a
H1

Fig. 1

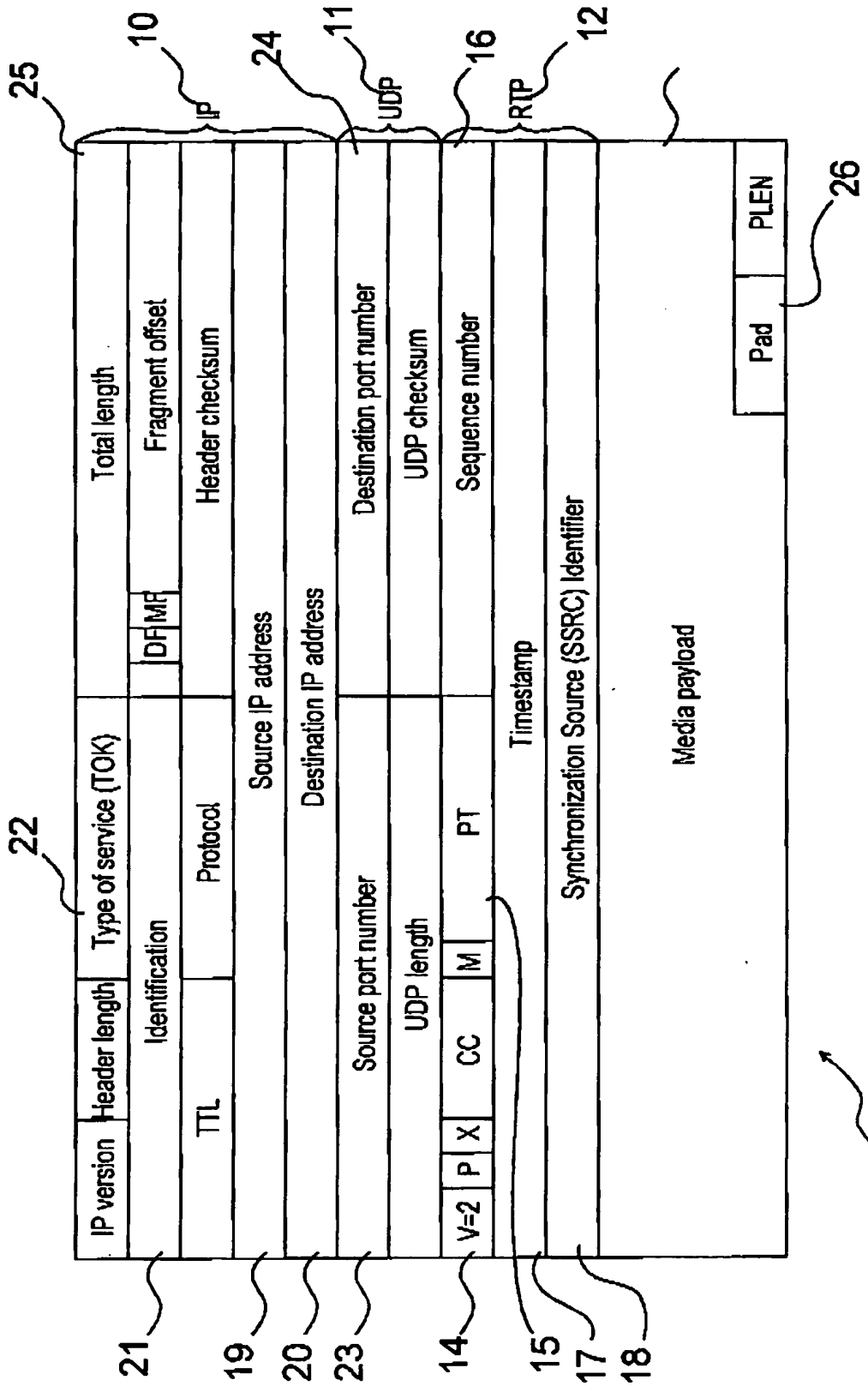
IP version
Header length
Type of service (TOK)
Total length
Identification
DF
MF
Fragment offset
TTL
Protocol
Header checksum
Source IP address
Destination IP address
Source port number
Destination port number
UDP length
UDP checksum
V=2
P
X
CC
M
PT
Sequence number
Timestamp
Synchronization Source (SSRC) Identifier
Media payload
Pad
PLEN
IP
UDP
RTP
10
11
12
16
22
24
25
26
3
14
15
17
18
19
20
21
23

Fig. 2

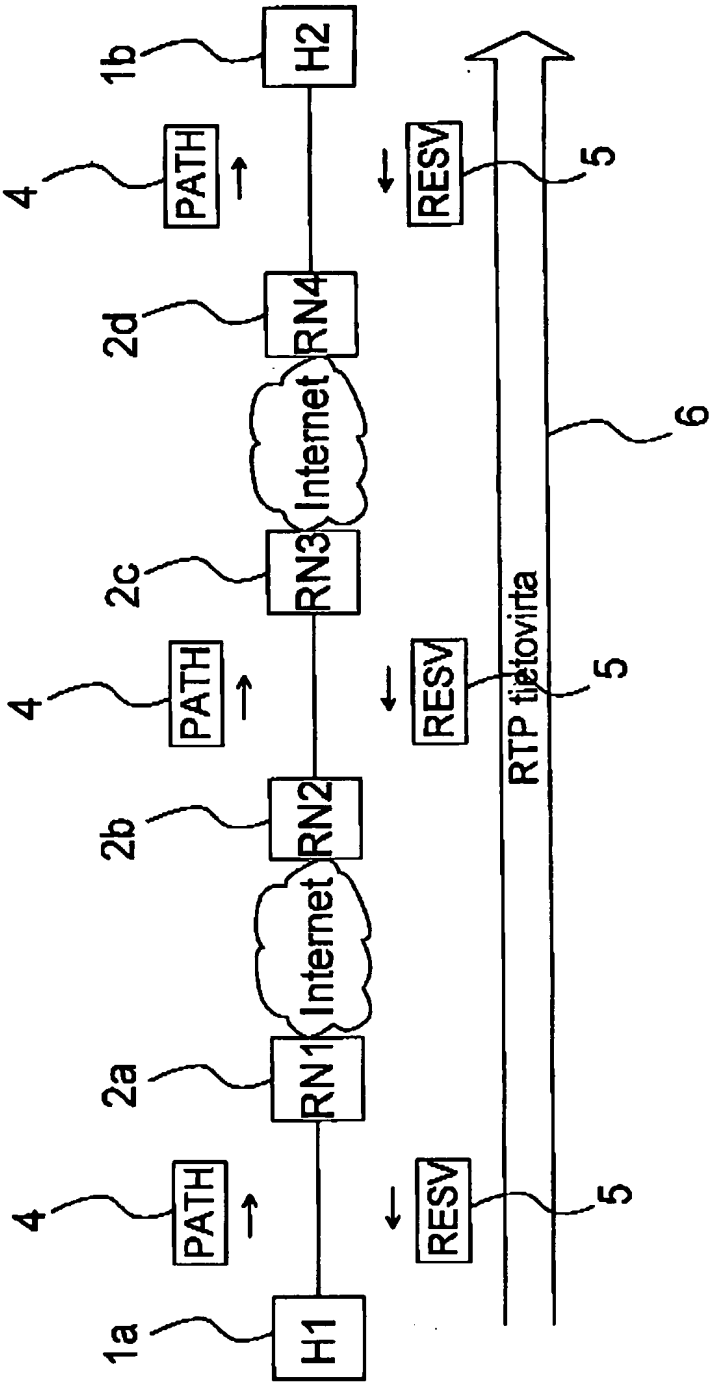
H1
H2
RN1
RN2
RN3
RN4
Internet
Internet
PATH
PATH
PATH
RESV
RESV
RESV
RTP tietovirta
1a
1b
2a
2b
2c
2d
4
4
4
5
5
5
6

Fig. 3

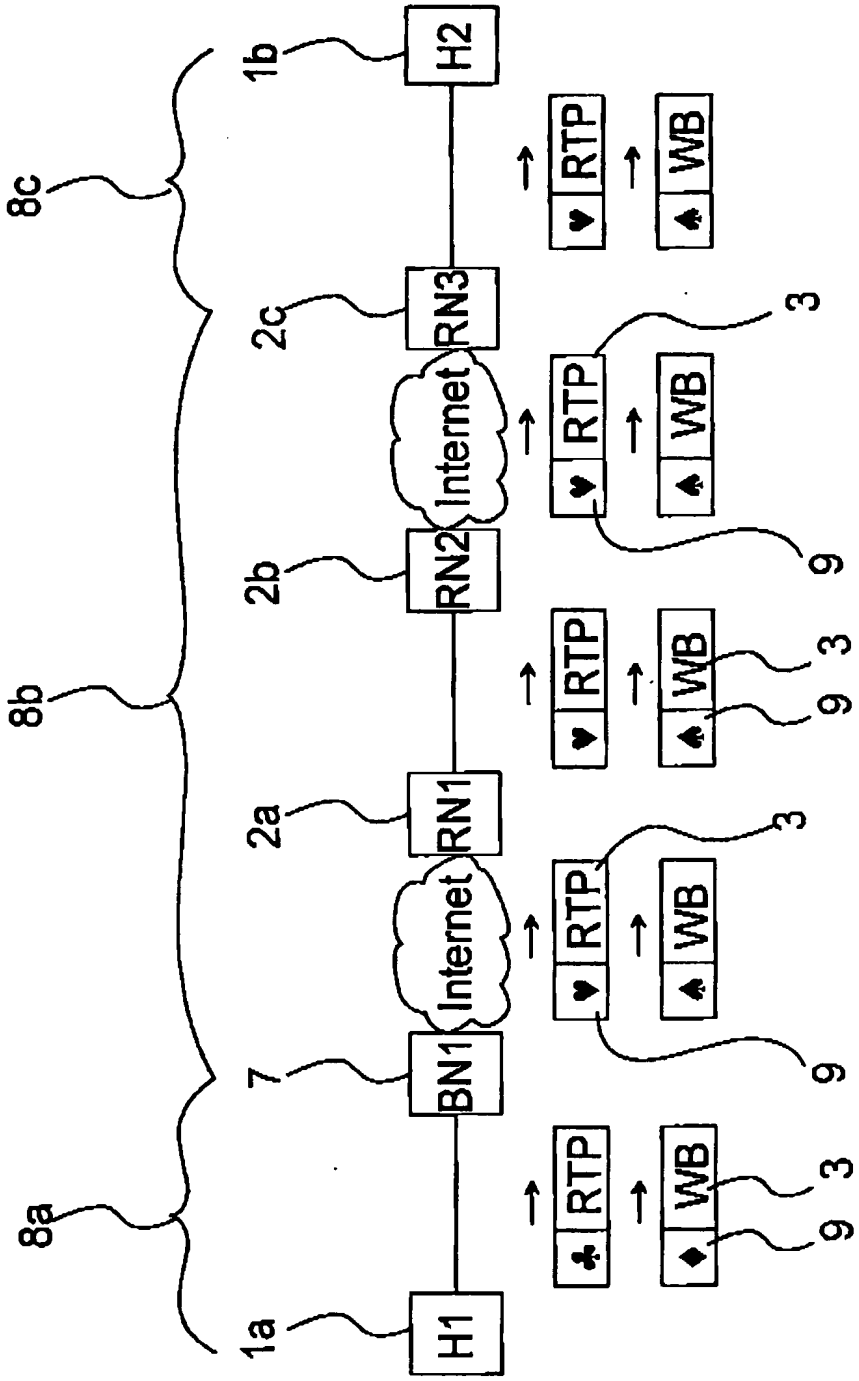
H1
H2
BN1
RN1
RN2
RN3
Internet
Internet
RTP
WB
1a
1b
2a
2b
2c
3
7
8a
8b
8c
9

Fig. 4